CLAIM AMENDMENTS

Claim Amendment Summary

Claims pending

- Before this Amendment: Claims 1-88.
- After this Amendment: Claims 1-2, 4-9, 11-18, 20-28, 30-41, 44-45, 47-62, 64-68, 70-80 and 82-90.

Non-Elected, Canceled, or Withdrawn claims: Claims 3, 10, 19, 29, 42, 43, 46, 63, 69 and 81.

Amended claims: Claims 1-2, 4-9, 11-18, 20-21, 24-28, 30-41, 44-45, 47-49, 52-62, 64-68, 71-77, 79-80 and 82-88.

New claims: Claims 89 and 90.

Claims:

1. (Currently Amended) A method comprising:

receiving a request, from an application at an application programming interface (API), to interact with a plurality of media; and

generating a media timeline based on the request, wherein the media timeline ~~that~~:

is exposed to the application ~~for exposure~~ via the API ~~to the application~~;

includes a plurality of nodes; and

defines a presentation of a first ~~said~~ media referenced by a first ~~said~~ node with respect to a second ~~said~~ media referenced by a second ~~said~~ node, wherein:

the first and second nodes are configured as parallel nodes such that the first node that is a child of a parent node is rendered concurrently with the second node that is a child of the same parent node; and

the media timeline is configured for dynamic creation such that at least one node is created while the media timeline is being rendered.

2. (Currently Amended) A method as described in claim 1, wherein one or more ~~said~~ nodes are configured as a sequence node such that one ~~said~~ node that is a child of the sequence node is rendered after another ~~said~~ node that is also a child of the sequence node.

3. (Canceled)

4. (Currently Amended) A method as described in claim 1, wherein one or more ~~said~~ nodes is configured as a root node that specifies a starting point for rendering the media timeline.

5. (Currently Amended) A method as described in claim 1, wherein the first and second said nodes reference the respective first and second ~~said~~ media utilizing respective first and second pointers.

6. (Currently Amended) A method as described in claim 1, wherein at least one ~~said~~ node includes metadata that describes rendering of the at least one ~~said~~ node.

7. (Currently Amended) A method as described in claim 6, wherein the metadata is selected from a group of metadata, the group of metadata comprising ~~consisting of~~:

a URL property for the media referenced by the at least one ~~said~~ node;

a source object property that specifies a source object which can resolve to a media source that provides the media referenced by the at least one ~~said~~ node;

a source object ID property that specifies a unique identifier of the source object;

a start time property that specifies when rendering of the at least one ~~said~~ node is to begin with respect to another ~~said~~ node;

a stop time property that specifies when rendering of the at least one ~~said~~ node is to stop with respect to another ~~said~~ node;

a media start property that specifies a time, during a duration of the media referenced by the at least one ~~said~~ node, that rendering of the media is to be started;

a media stop property that specifies a time, during a duration of the media referenced by the at least one ~~said~~ node, that rendering of the media is to be stopped;

a time format property that specifies a time format for at least one of the start time property, the stop time property, the media start property, and the media stop property;

a stream selection property which specifies one of a plurality of streams for rendering of the media referenced by the at least one ~~said~~ node;

a format based property that specifies a format for the media referenced by the at least one ~~said~~ node;

a loop count property that specifies a number of times the at least one ~~said~~ node is to be rendered;

a disabled property that specifies whether the at least one ~~said~~ node is to be rendered when the media timeline is rendered;

a generic property that serves as a repository of information related to the at least one ~~said~~ node, wherein the generic property is configured for

specification by at least one of the application and a timeline source for rendering the media timeline;

a noskip property that specifies that the rendering of the at least one ~~said~~ node is not to be skipped when the media timeline is rendered; and

a noskip child property that specifies that the at least one ~~said~~ node has another ~~said~~ node, which is a child of the at least one ~~said~~ node, which specifies that the rendering of the other ~~said~~ node is not to be skipped when the media timeline is rendered.

8. (Currently Amended) A method as described in claim 1, wherein at least one ~~said~~ node is configured to reference an effect to be applied to an output of media referenced by the node.

9. (Currently Amended) A method as described in claim 1, wherein the media timeline is configured for dynamic loading such that metadata included in at least one ~~said~~ node specifies a collection of ~~said~~ nodes to be loaded when the media timeline is rendered.

10. (Canceled)

11. (Currently Amended) A method as described in claim 1, wherein at least one ~~said~~ node is specified as read-only.

12. (Currently Amended) A method as described in claim 1, wherein at least one ~~said~~ node is configured for communication of events to another ~~said~~ node such that a change may be made to the media timeline while the media timeline is rendered.

13. (Currently Amended) A method as described in claim 1, wherein the first and second ~~said~~ media have different formats~~, one to another~~.

14. (Currently Amended) One or more computer readable media ~~comprising~~ storing computer executable instructions that, when executed by a computer, direct the computer to perform the method of claim 1.

15. (Currently Amended) A method comprising:

generating a media timeline by an application, wherein the media timeline:

includes a plurality of nodes;

defines a presentation of a first ~~said~~ media referenced by a first ~~said~~ node with respect to a second ~~said~~ media referenced by a second ~~said~~ node; ~~and~~

passing the media timeline to a timeline source for rendering; and

is configured for dynamic creation such that at least one node is created while the media timeline is being rendered.

16. (Currently Amended) A method as described in claim 15, wherein the first and second ~~said~~ media have different formats~~, one to another~~.

17. (Currently Amended) A method as described in claim 15, wherein at least one ~~said~~ node is configured to reference an effect to be applied to an output of media referenced by the node.

18. (Currently Amended) A method as described in claim 15, wherein the media timeline is configured for dynamic loading such that metadata included in at least one ~~said~~ node specifies a collection of ~~said~~ nodes to be loaded when the media timeline is rendered.

19. (Canceled)

20. (Currently Amended) One or more computer readable media ~~comprising~~ storing computer executable instructions that, when executed by a computer, direct the computer to perform the method of claim 15.

21. (Currently Amended) A method comprising:

specifying an effect to be applied to one or more of a plurality of media when the media is rendered; and

generating a media timeline configured for exposure via an application programming interface (API), wherein:

the media timeline includes a plurality of nodes;

two or more ~~said~~ of the plurality of nodes reference respective ~~said~~ media; ~~and~~

one or more ~~said~~ of the plurality of nodes that reference the one or more ~~said~~ of the plurality of media include metadata that describes the effect; and

the media timeline is configured for dynamic creation such that at least one node is created while the media timeline is rendered.

22. (Original) A method as described in claim 21, wherein the effect is a simple effect provided by a software component that is configured to:

receive a single stream of media;

apply the effect to the single stream; and

output the applied single stream.

23. (Original) A method as described in claim 21, wherein the effect is a composite effect provided by a software component that is configured to:

receive at least two streams of media;

apply the effect to the at least two streams; and

output a single stream of media composed of the applied at least two streams.

24. (Currently Amended) A method as described in claim 21, wherein the effect is a composite effect provided by a software component that is configured to ~~at least one of~~ analyze at least two streams of media ~~and~~ or output at least two streams of media.

25. (Currently Amended) A method as described in claim 21, wherein the effect is a transition effect to be applied as a transition from a first ~~said~~ media referenced by a first ~~said~~ node to a second ~~said~~ media referenced by a second ~~said~~ node.

26. (Currently Amended) A method as described in claim 21, wherein the effect includes metadata that describes the effect that is selected from the group of metadata, the group of metadata comprising ~~consisting of~~:

an effect object GUID property that specifies a GUID to be used to create a transform object that is configured to provide the effect;

an effect object property that references an effect object that is configured to provide the effect;

a priority property that specifies an ordering of a plurality of ~~said~~ effects, one to another;

a start time property that specifies when processing of the effect is to begin with respect to rendering ~~of the~~ one of the ~~or more said~~ nodes;

a stop time property that specifies when processing of the effect is to stop with respect to rendering ~~of the~~ one of the ~~or more said~~ nodes;

a time format property that specifies a time format for at least one of the start time property and the stop time property

a number of inputs property that specifies a number of inputs to the effect;

a number of outputs property that specifies a number of outputs from the effect;

an output major type property that specifies a major type for media, to which, the effect is to be applied; and

an input connections property that specifies the one or more ~~said~~ nodes that are to be processed by the effect.

27. (Currently Amended) A method as described in claim 21, wherein at least one ~~said~~ node includes metadata that describes rendering of the at least one ~~said~~ node.

28. (Currently Amended) A method as described in claim 21, wherein the media timeline is configured for dynamic loading such that metadata included in at least one ~~said~~ node specifies a collection of ~~said~~ nodes to be loaded when the media timeline is rendered.

29. (Canceled)

30. (Currently Amended) A method as described in claim 21, wherein at least one ~~said~~ node is specified as read-only.

31. (Currently Amended) A method as described in claim 21, wherein at least one ~~said~~ node is configured for communication of events to another ~~said~~ node such that a change may be made to the media timeline while the media timeline is rendered.

32. (Currently Amended) One or more computer readable media ~~comprising~~ storing computer executable instructions that, when executed by a computer, cause the computer to perform the method of claim 21.

33. (Currently Amended) In a media timeline exposed via an application programming interface and having a plurality of nodes, a method comprising:

rendering a first media item referenced by a first ~~said~~ node;

receiving a call for a second ~~said~~ node that references a second media item; and

creating the second ~~said~~ node while rendering the first media item.

34. (Currently Amended) A method as described in claim 33, further comprising rendering a second media item referenced by the second ~~said~~ node when the rendering of the first media item is completed.

35. (Currently Amended) A method as described in claim 33, further comprising:

rendering the second media item referenced by the second ~~said~~ node;

receiving a call for a third ~~said~~ node that references a third media item; and

creating the third ~~said~~ node.

36. (Currently Amended) A method as described in claim 33, wherein the media timeline is configured for dynamic loading such that metadata included in at least one ~~said~~ node specifies a collection of ~~said~~ nodes to be loaded when the media timeline is rendered.

37. (Currently Amended) A method as described in claim 33, wherein at least one ~~said~~ node is configured to reference an effect to be applied to an output of media referenced by the node.

38. (Currently Amended) A method as described in claim 33, wherein at least one ~~said~~ node is specified as read-only.

39. (Currently Amended) A method as described in claim 33, wherein at least one ~~said~~ node is configured for communication of events to another ~~said~~ node such that a change may be made to the media timeline while the media timeline is rendered.

40. (Currently Amended) One or more computer readable media ~~comprising~~ storing computer executable instructions that, when executed by a computer, direct the computer to perform the method of claim 33.

41. (Currently Amended) In a media timeline exposed via an application programming interface, the media timeline having a plurality of nodes, at least two ~~said~~ of which ~~nodes~~ reference ~~referencing~~ respective media, one or more ~~said~~ nodes each having metadata that references a node grouping, a method comprising:

loading a first ~~said~~ node for rendering;

examining metadata associated with the first ~~said~~ node to determine a first ~~said~~ node grouping to be loaded in conjunction with the first node;

loading each ~~said~~ node referenced by the first ~~said~~ node grouping;

rendering the first ~~said~~ node grouping;

examining at least one node ~~one or more said nodes~~ in the first ~~said~~ node grouping to determine a second ~~said~~ node grouping, wherein the examining at least one node in the first node grouping is performed during the rendering of the first node grouping;

loading each ~~said~~ node referenced by the second ~~said~~ node grouping; and

rendering the second node grouping when the rendering of the first ~~said~~ node grouping is completed, ~~rendering the second said node grouping~~ wherein:

the media timeline is configured for dynamic creation where one node is created while the media timeline is being rendered, the dynamic creation of the one node being performed by a node source that includes

data that defines properties and interrelationships of the created node; and

at least one node is configured for communication of an event to another node such that a change may be made to the media timeline while the media timeline is being rendered, wherein the plurality of nodes of the media timeline that are affected by the change are automatically updated.

42. (Canceled)

43. (Canceled)

44. (Currently Amended) A method as described in claim 41, wherein at least one ~~said~~ node is configured to reference an effect to be applied to an output of media referenced by the node.

45. (Currently Amended) A method as described in claim 41, wherein at least one ~~said~~ node is specified as read-only.

46. (Canceled)

47. (Currently Amended) A method as described in claim 41, wherein a first ~~said~~ node references ~~said~~ media having a different format than ~~said~~ media referenced by a second ~~said~~ node.

48. (Currently Amended) One or more computer readable media ~~comprising~~ storing computer executable instructions that, when executed by a computer, direct the computer to perform the method of claim 41.

49. (Currently Amended) In a media timeline exposed via an application programming interface (API), the media timeline having a plurality of nodes, two or more ~~said~~ nodes each referencing respective media, a method comprising:

rendering a first ~~said~~ node to output a referenced first said media;

during the rendering, changing one or more properties of a second ~~said~~ node; and

initiating, by an event generator located on the second ~~said~~ node, an event for communication to a parent ~~said~~ node of the second ~~said~~ node, wherein the event describes the changing.

50. (Original) A method as described in claim 49, wherein the event is communicated to at least one of an application over the API and a timeline source for rendering the media timeline.

51. (Original) A method as described in claim 49, wherein the one or more properties are selected from the group consisting of:

node added event;

node removed event;

node changing event;

node changed event;

remove children event;

node source added event;

node source removed event;

node sort event; and

node moved event.

52. (Currently Amended) A method as described in claim 49, wherein:

one <u>at least one</u> ~~said~~ node of the media timeline is configured as a root node; and

each ~~said~~ event generated by one of the plurality of nodes that is a child of the root node is communicated to the root node.

53. (Currently Amended) A method as described in claim 49, wherein the media timeline is configured for dynamic loading such that metadata included in at least one ~~said~~ node specifies a collection of ~~said~~ nodes to be loaded when the media timeline is rendered.

54. (Currently Amended) A method as described in claim 49, wherein the media timeline is configured for dynamic creation such that at least one ~~said~~ node is created while the media timeline is rendered.

55. (Currently Amended) A method as described in claim 49, wherein at least one ~~said~~ node is specified as read-only.

56. (Currently Amended) A method as described in claim 49, wherein at least one ~~said~~ node is configured to reference an effect to be applied to an output of ~~said~~ media referenced by the node.

57. (Currently Amended) One or more computer readable media ~~comprising~~ storing computer executable instructions that, when executed by a computer, direct the computer to perform the method of claim 49.

58. (Currently Amended) An application programming interface embodied on a computer storage medium, which when interfaced with a computer, exposes ~~for exposing~~ a media timeline to one or more independent applications, ~~and~~ the application programming interface comprising:

the media timeline ~~including~~ comprising a plurality of nodes ~~that are~~ callable by one ~~said~~ application, wherein:

each ~~said~~ node includes metadata that describes the node;

one or more ~~said~~ nodes reference a corresponding media item;

the plurality of nodes are arranged in a tree structure; and

the arrangement of the plurality of nodes, one to another, describes an order for rendering the plurality of nodes, wherein the media timeline is configured for dynamic creation such that at least one node is created while the media timeline is rendered.

59. (Currently Amended) An application programming interface as described in claim 58, wherein the metadata for each ~~said~~ node is selected from a group of metadata, the group of metadata comprising ~~consisting of~~:

a URL property for the media referenced by the node;

a source object property that specifies a source object which can resolve to a media source that provides the media referenced by the node;

a source object ID property that specifies a unique identifier of the source object;

a start time property that specifies when rendering of the node is to begin with respect to another ~~said~~ node;

a stop time property that specifies when rendering of the node is to stop with respect to another ~~said~~ node;

a media start property that specifies a time, during a duration of the media referenced by at the node, that rendering of the media is to be started;

a media stop property that specifies a time, during a duration of the media referenced by the node, that rendering of the media is to be stopped;

a time format property that specifies a time format for at least one of the start time property, the stop time property, the media start property, and the media stop property;

a stream selection property which specifies one of a plurality of streams for rendering of the media referenced by the node;

a format based property that specifies a format for the media referenced by the node;

a loop count property that specifies a number of times the node is to be rendered;

a disabled property that specifies whether the node is to be rendered when the media timeline is rendered;

a noskip property that specifies that the rendering of the node is not to be skipped when the media timeline is rendered; and

a noskip child property that specifies that the node has another ~~said~~ node, which is a child of the node, which specifies that the rendering of the other ~~said~~ node is not to be skipped when the media timeline is rendered.

60. (Currently Amended) An application programming interface as described in claim 58, wherein at least one ~~said~~ node is configured to reference an effect to be applied to an output of media referenced by the node.

61. (Currently Amended) An application programming interface as described in claim 58, wherein at least one ~~said~~ node includes metadata that describes rendering of the at least one ~~said~~ node.

62. (Currently Amended) An application programming interface as described in claim 58, wherein the media timeline is configured for dynamic loading such that metadata included in at least one ~~said~~ node specifies a collection of ~~said~~ nodes to be loaded when the media timeline is rendered.

63. (Canceled)

64. (Currently Amended) An application programming interface as described in claim 58, wherein at least one ~~said~~ node is specified as read-only.

65. (Currently Amended) An application programming interface as described in claim 58, wherein at least one ~~said~~ node is configured for communication of events to another ~~said~~ node such that a change may be made to the media timeline while the media timeline is rendered.

66. (Currently Amended) An application programming interface ~~for~~ stored on a computer storage medium, that when accessed by a computer facilitates acts comprising:

exposing a media timeline to one or more independent applications, ~~and comprising~~ the media timeline ~~including~~ comprising a plurality of nodes ~~that are~~ callable by one ~~said~~ application, wherein:

two or more ~~said~~ of the nodes reference respective media;

the plurality of nodes are arranged in a hierarchy to include a parent ~~said~~ node and a child ~~said~~ node; and

the child ~~said~~ node is configured for initiating an event for communication to the parent ~~said~~ node, wherein the event:

is configured such that a change may be made to one or more properties of the child node while the media timeline is rendered; and

describes the change.

67. (Currently Amended) An application programming interface as described in claim 66, wherein another ~~said~~ node, which is not a parent of the child ~~said~~ node, subscribes to the child ~~said~~ node to receive the event.

68. (Currently Amended) An application programming interface as described in claim 66, wherein another ~~said~~ node subscribes to the child ~~said~~ node to receive:

the event initiated by the child ~~said~~ node; and

one or more ~~said~~ events initiated by children of the child ~~said~~ node.

69. (Canceled)

70. (Original) An application programming interface as described in claim 66, wherein the event describes a change made to the media timeline, the event selected from the group consisting of:

node added event;

node removed event;

node changing event;

node changed event;

remove children event;

node source added event;

node source removed event;

node sort event; and

node moved event.

71. (Currently Amended) An application programming interface as described in claim 66, wherein:

one ~~said~~ node of the media timeline is configured as a root node; and

each ~~said~~ event generated by one of the plurality of nodes that is a child of the root node is communicated to the root node.

72. (Currently Amended) An application programming interface as described in claim 66, wherein the media timeline is configured for dynamic loading such that metadata included in at least one ~~said~~ node specifies a collection of ~~said~~ nodes to be loaded when the media timeline is rendered.

73. (Currently Amended) An application programming interface as described in claim 66, wherein the media timeline is configured for dynamic creation such that at least one ~~said~~ node is created while the media timeline is rendered.

74. (Currently Amended) An application programming interface as described in claim 66, wherein at least one ~~said~~ node is configured to reference an effect to be applied to an output of media referenced by the node.

75. (Currently Amended) An application programming interface as described in claim 66, wherein at least one ~~said~~ node is specified as read-only.

76. (Currently Amended) An application programming interface ~~for~~ embodied in an infrastructure layer of a computer that, when interacted with by an application facilitates actions comprising:

exposing a media timeline comprising one or more nodes to the application; ~~one or more independent applications,~~ and

enabling the application to call any of the one or more nodes, wherein each of the ~~comprising the media timeline including a plurality of nodes that are callable by one said application, wherein~~ one or more ~~said~~ nodes:

references ~~reference~~ corresponding media;

includes ~~include~~ metadata describing one or more properties for rendering the corresponding media; and

includes ~~include~~ metadata specifying the node as read-only; and

configuring the media timeline for dynamic creation such that at least one of the one or more nodes is created while the media timeline is being rendered.

77. (Currently Amended) A system comprising:

a plurality of media;

a plurality of applications; and

an infrastructure layer that:

provides an application programming interface (API) for interaction by the plurality of applications with the plurality of media when any ~~said~~ application is executed; and

exposes a media timeline, callable by the plurality of applications via the API upon an execution thereof, and that defines a presentation of the plurality of media, wherein the media timeline:

includes a plurality of nodes that each reference respective media; and

is configured for dynamic creation such that at least one node is created while the media timeline is rendered.

78. (Original) A system as described in claim 77, wherein the media timeline is configured to reference an effect for application to an output of one or more of the plurality of media.

79. (Currently Amended) A system as described in claim 77, wherein:

~~the media timeline includes a plurality of nodes;~~

~~one or more said nodes references respective said media;~~

the media timeline defines a presentation of a first ~~said~~ media referenced by a first ~~said~~ node with respect to a second ~~said~~ media referenced by a second ~~said~~ node; and

at least one ~~said~~ node includes metadata that describes rendering of the at least one ~~said~~ node.

80. (Currently Amended) A system as described in claim 77, wherein[[:]]

~~the media timeline includes a plurality of nodes;~~

~~one or more said nodes references respective said media; and~~

the media timeline is configured for dynamic loading such that metadata included in at least one ~~said~~ node specifies a collection of ~~said~~ nodes to be loaded when the media timeline is rendered.

81. (Canceled)

82. (Currently Amended) A system as described in claim 77, wherein~~:~~

~~the media timeline includes a plurality of nodes;~~

~~one or more said nodes references respective said media; and~~

at least one ~~said~~ node is specified as read-only.

83. (Currently Amended) A system as described in claim 77, wherein~~:~~

~~the media timeline includes a plurality of nodes;~~

~~one or more said nodes references respective said media; and~~

at least one said node is configured for communication of events to another ~~said~~ node such that a change may be made to the media timeline while the media timeline is rendered.

84. (Currently Amended) A computer comprising:

a processor; and

memory configured to maintain:

a plurality of media;

a plurality of applications, wherein each ~~said~~ application is configured to request at least one of editing, encoding, and rendering of the plurality of media; and

an infrastructure layer configured to:

provide an application programming interface (API) for interaction by the plurality of applications with the plurality of media; and

expose a media timeline, callable by the plurality of applications via the API, that includes a plurality of nodes that define a presentation of the plurality of media, wherein the media timeline specifies:

delayed creation of one or more ~~said~~ nodes when the media timeline is rendered;

metadata that is utilized by the plurality of applications, wherein the metadata describes:

how rendering of the plurality of nodes is to be initiated;

properties of the plurality of nodes; and

node types of the plurality of nodes;

at least one node that is configured for communication of events to another node such that a change may be made to the media timeline while the media timeline is being rendered; and

at least one node that is a parallel node that provides simultaneous rendering of at least two child nodes.

85. (Currently Amended) A computer as described in claim 84, wherein the delayed creation further comprises creating the one or more ~~said~~ nodes when called by one or more ~~said~~ applications.

86. (Currently Amended) A computer as described in claim 84, wherein the media timeline is configured for dynamic loading such that metadata included in at least one ~~said~~ node specifies a collection of ~~said~~ nodes to be loaded when the media timeline is rendered.

87. (Currently Amended) A computer as described in claim 84, wherein at least one ~~said~~ node is configured to reference an effect to be applied to an output of media referenced by the node.

88. (Currently Amended) A computer as described in claim 84, wherein at least one ~~said~~ node of the media timeline is configured for communication of events to another ~~said~~ node such that a change may be made to the media timeline while the media timeline is rendered.

89. (New) A method as described in claim 33, wherein the first node and the second node are each selected from a group of node types, the group of node types comprising:

a root node that specifies a starting point for rendering the media timeline, the root node including metadata that describes how rendering is to be initiated;

a leaf node that directly maps to media to be rendered and output, the leaf node including metadata that describes how to retrieve the media;

a sequence node that includes metadata that describes a rendering order of a plurality of leaf nodes to the sequence node; and

a parallel node that includes metadata specifying a plurality of leaf nodes that are rendered simultaneously.

90. (New) A method as described in claim 41, wherein the first node is selected from a group of node types, the group of node types comprising:

a root node that specifies a starting point for rendering the media timeline, the root node including metadata that describes how rendering is to be initiated;

a sequence node that includes metadata that describes a rendering order of a plurality of leaf nodes to the sequence node; and

a parallel node that includes metadata specifying a plurality of leaf nodes that are rendered simultaneously.